Network address translation in a gateway

CPOL No.: 79361 **Seq No.:** 2899 **Status:** Pending **Submitted:** **Modified:**

Idea Details

Appl. No.: 09/910,937
Exhibit A (Page 1of 1)

Inventors:

Anuradha Karuppiah (ak) *Employee has left Cisco.*

Amit Phadnis (asp) *Employee has left Cisco.*

Praneet Bachheti (praneetb) *Employee has left Cisco.*

Background: SSG (Service Selection Gateway) provides subscribers access to multiple overlapping services. For eg. Service1 an Service2 accessible via the SSG can have overlapping adresses. Separate routing tables are maintained for each of these service domains (MFIB approach).

Each user can access multiple services simultaneously each of which will assign a different real IP address to the user, hence requiring NAT.

The services can be overlapping, so can the NAT entries created. Since the NAT entries go into a global table we require a service specific distinguisher. The DC branch maintained for SSG provides service IDB based NAT approach, where all the NAT entries are placed in a single global table and distinguished with an additional key - the uplink/service IDB. This however is not available on IOS mainline.

By having this additional key for a per-packet NAT lookup, we further degrade data forwarding performance.

Assuming that 1000 users connect to 10 services each via SSG, we have 10,000 connections and consequently 10,000 NAT entries, which all go into a single, global table. The NAT table is a hash table which uses linear probing to avoid collision (within a single hash node). By overloading a single table with entrie we increase the probability of multiple entries against a single hash node, hence requiring a more extensive linear search.

Besides locating a NAT entry involves an additional interface matching along with the normal address/port matching.

Prior Art: ---

Summary: The solution is a per-service NAT table - MNAT. A particular service will be bound to a NAT table (there is no additional processing for locating the NAT table on a per-packet basis). A NAT table in this case will contain only limited entries, specific to its service domain.

Let's consider the same eg. of 1000 users connecting to 10 services each. We woul have 10 NAT tables, each having just 1000 entries.

Advantages: - Having a smaller service based NAT table, as compared to a single global one, avoids the probability of overloading a hash node with multiple NAT entries. Henc the NAT lookup would definitly be faster as linear checks are reduced.

- We also avoid the use of an additional service idb distinguisher, hence reducir the per-packet NAT lookup processing.